PROYECTO FINAL - RECONOCIMIENTO DE ACTIVIDADES FÍSICAS

Introducción

En este proyecto se va a aplicar todo lo aprendido en la asignatura Modelos de Regresión para construir un algoritmo capaz de predecir si el usuario está caminando, subiendo escaleras o quieto.

Para ello, se tomarán como datos brutos las señales del acelerómetro de nuestro móvil para cada una de las actividades, se realizará un análisis exploratorio de datos, se producirá un set de datos integrado con toda la clase, se elaborará un modelo de regresión logística y se extraerán conclusiones, así como plantear ideas de mejora del modelo.

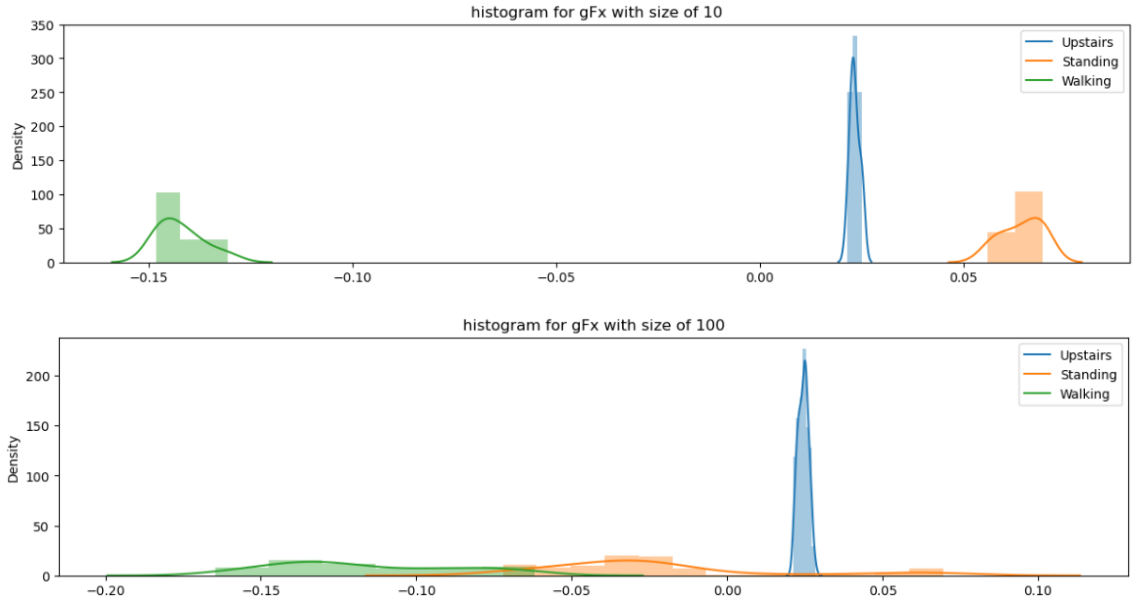
Parte I – Con el set de datos individual

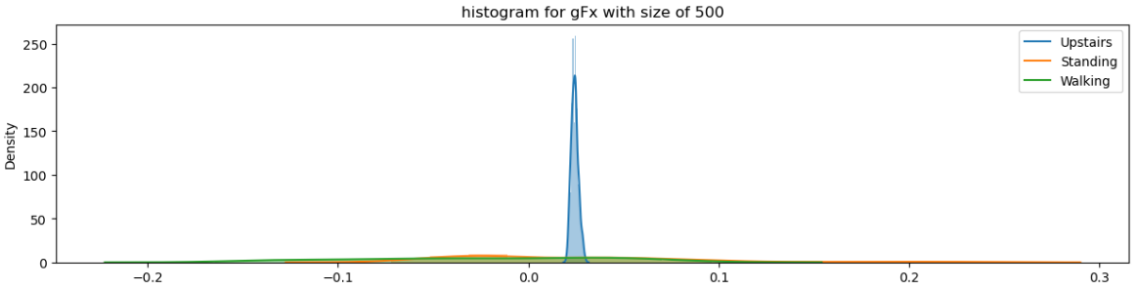
Realiza un Análisis Exploratorio de Datos (EDA)

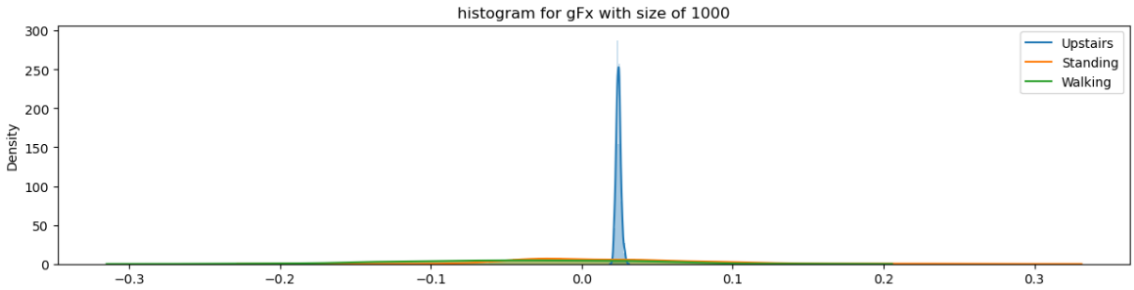
1. EDA de los datos "*standing*": EDA_Standing.ipynb
2. EDA de los datos "*walking*": EDA_Walking.ipynb
3. Eda de los datos "*stairs*": EDA_Stairs.ipynb

Determina el tamaño de muestreo adecuado para obtener parámetros estadísticos con capacidad predictiva

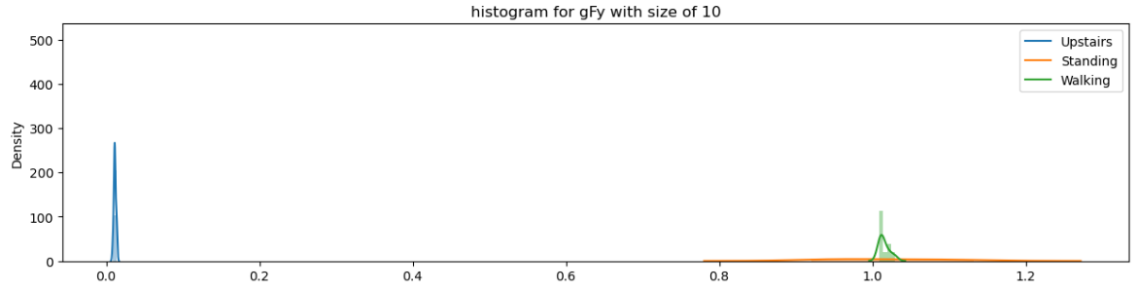
Eje X

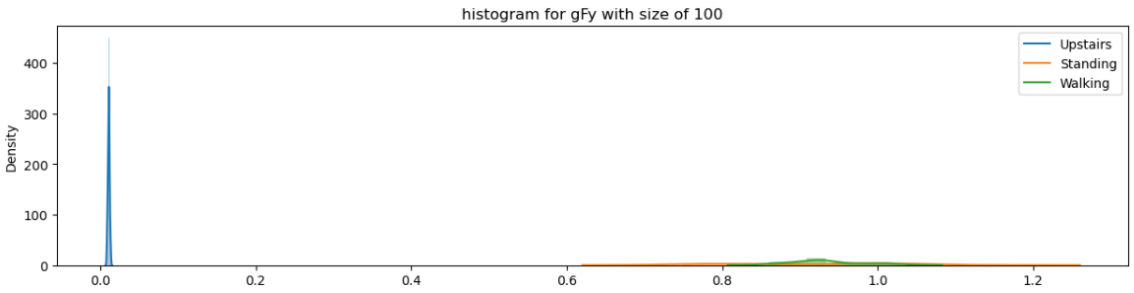


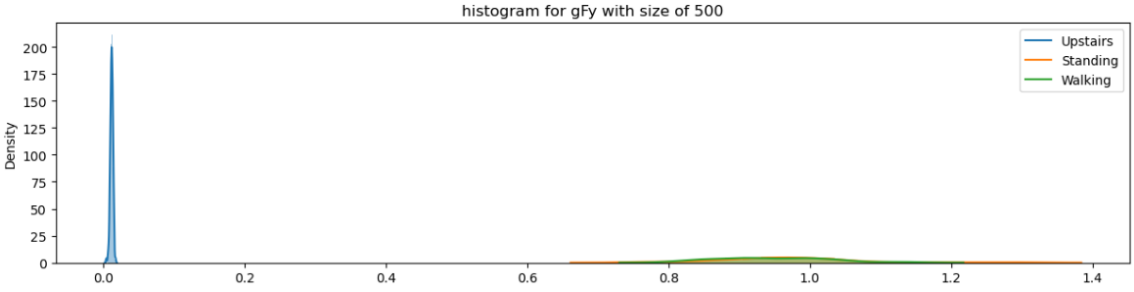

histogram for gFx with size of 500
Upstairs
Standing
Walking
Density
250
200
150
100
50
0
−0.2
−0.1
0.0
0.1
0.2
0.3

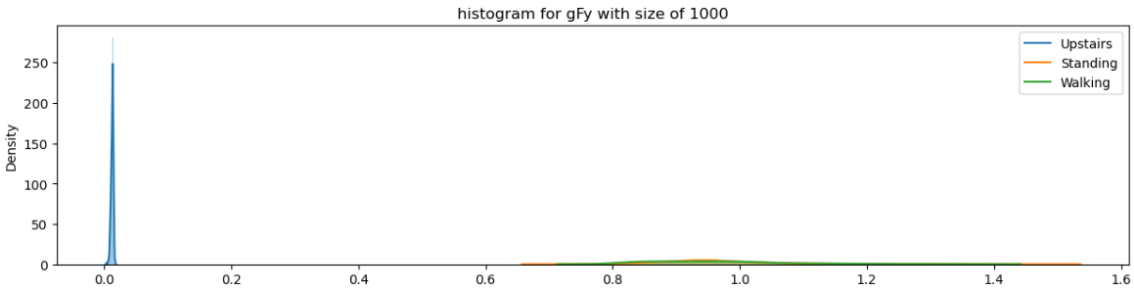

histogram for gFx with size of 1000
Upstairs
Standing
Walking
Density
300
250
200
150
100
50
0
−0.3
−0.2
−0.1
0.0
0.1
0.2
0.3

Eje Y

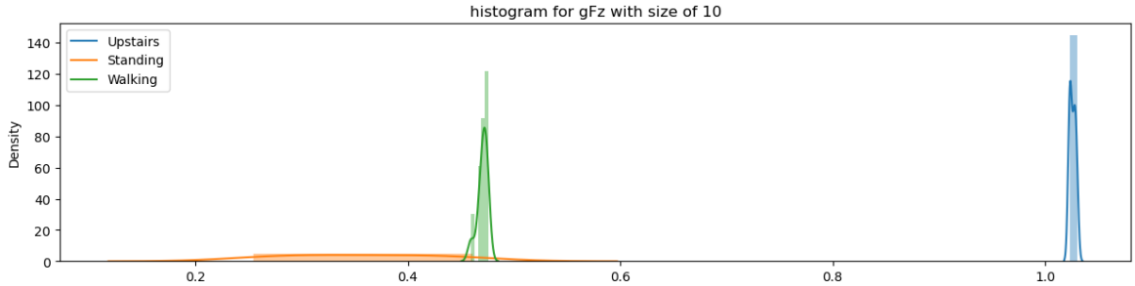

histogram for gFy with size of 10
Upstairs
Standing
Walking
Density
500
400
300
200
100
0
0.0
0.2
0.4
0.6
0.8
1.0
1.2

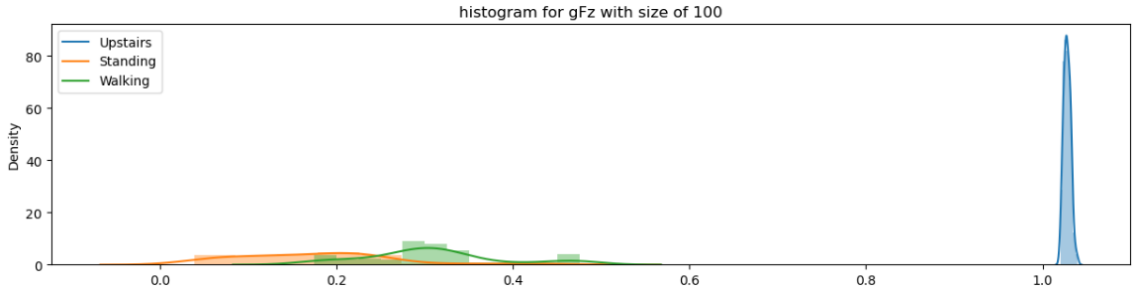

histogram for gFy with size of 100
Upstairs
Standing
Walking
Density
400
300
200
100
0
0.0
0.2
0.4
0.6
0.8
1.0
1.2

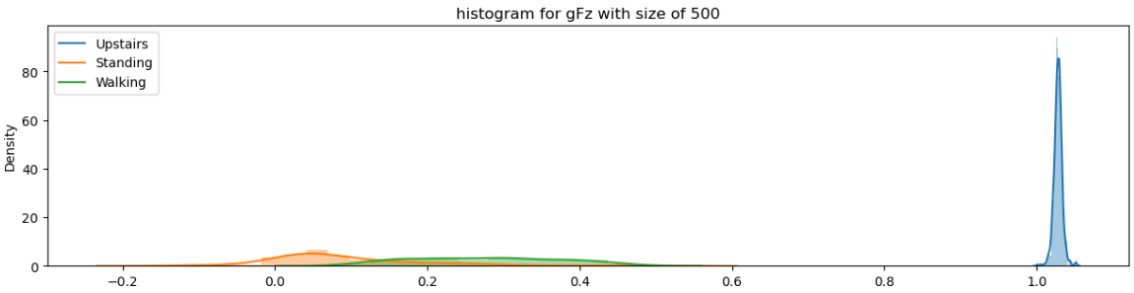

histogram for gFy with size of 500
Upstairs
Standing
Walking
Density
200
175
150
125
100
75
50
25
0
0.0
0.2
0.4
0.6
0.8
1.0
1.2
1.4

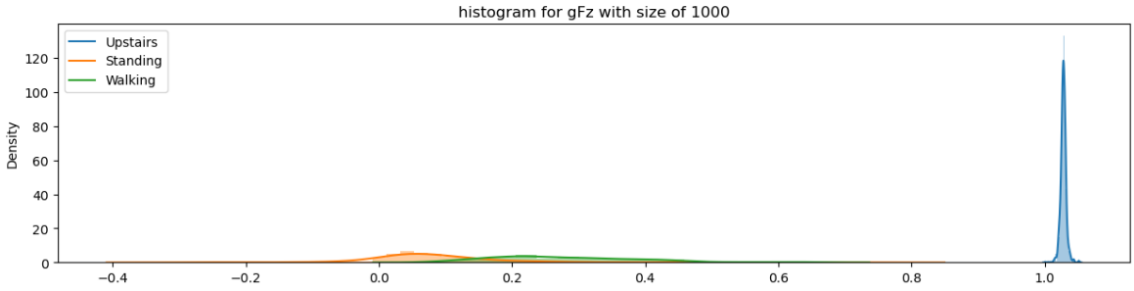


Eje Z









Conclusiones

Como se puede ver en las gráficas superiores, generadas en el archivo EDA_Total.ipynb, es en torno a las 500 muestras cuando las distribuciones de densidad para cada eje se estabilizan, indicando que éstas son representativas de la población. Por otra parte, para este tamaño muestral se ven bien las diferencias en la distribución de cada eje entre las diferentes actividades.

Por ende, un tamaño de aproximadamente 500 muestras es suficientemente adecuado para obtener parámetros estadísticos con capacidad predictiva.

Añade variables adicionales para aumentar la capacidad predictiva

Variables añadidas:

- **Media móvil**: indicador utilizado para suavizar los datos de una serie temporal, eliminando el ruido y mostrando una tendencia subyacente más clara. Se calcula tomando un promedio de un cierto número de períodos de tiempo consecutivos, que se desplaza a medida que se reciben nuevos datos.
- **Energía/varianza móvil**: indicador que mide la magnitud de los cambios en una serie temporal, se calcula como el promedio de la varianza de un cierto número de períodos de tiempo consecutivos, que se desplaza a medida que se reciben nuevos datos.
- **Mediana móvil**: indicador que se utiliza para suavizar los datos de una serie temporal, similar a la media móvil, pero en lugar de utilizar el promedio, se utiliza la mediana.
- **Curtosis móvil**: indicador que se emplea para medir la concentración de los datos alrededor del valor medio, se calcula tomando la curtosis de un cierto número de períodos de tiempo consecutivos, que se desplaza a medida que se reciben nuevos datos.

Parte II – Elaboración del set de datos integrado

Integración de las series temporales individuales en un único set de datos

Ver el archivo DatasetCreation.ipynb, donde se crea el set de datos integrado en crudo (sin las variables añadidas aún) combined_csv_without_features.csv. Se añaden las columnas 'sujeto', 'móvil' y 'actividad'.

Adición de las variables propuestas por el enunciado y otras variables extras

Ver la primera parte del archivo FeatureEngineering.ipynb, donde se calculan las medias, varianzas, medianas y curtosis móviles para 400 periodos con un step de 10 periodos (inicialmente eran 20 periodos, pero se decidió cambiarlo a 10 para tener más datos). Esto resulta en la creación del archivo final combined_and_transformed_data.csv

Separación de los datos en *train* y *test* sets

Ver la última parte del archivo FeatureEngineering.ipynb, donde se separan los datos en train (13 usuarios, aprox. 80%) y test (3, usuarios, aprox. 20%) sets. Esto resulta en la creación de los archivos train_data.csv y test_data.csv

Parte III – Con el set de datos conjunto

Entrena individualmente un modelo de Regresión Logística

Se entrenó un modelo de Regresión Logística con las siguientes características:

- Las ventanas de agrupación recogen 400 periodos con un desfase de 10 periodos (elegido en conjunto por la clase como la mejor opción de todas las probadas).
- Las siguientes variables fueron elegidas teniendo en cuenta su correlación con la actividad: gFx_mean, gFy_mean, TgF_mean, gFx_energia, gFy_energia, gFz_energia, TgF_energia, gFx_median, gFy_median, TgF_median, gFx_kurtosis, gFz_kurtosis.

- Los siguientes parámetros fueron configurados en el modelo:

```
LogisticRegression(max_iter=500,
                   random_state=0,
                   penalty='none',
                   solver = 'lbfgs',
                   class_weight='balanced')
```

  - *max_iter*: es el número máximo de iteraciones que se realizarán durante el proceso de entrenamiento del modelo. A mayor número de iteraciones, el modelo se ajustará mejor a los datos de entrenamiento, pero hay peligro de overfitting. Por ello, lo se ha puesto demasiado alto.
  - *random_state*: es un valor numérico utilizado para inicializar el generador de números aleatorios. Se especifica un valor para que los resultados sean reproducibles.
  - *penalty*: es el parámetro que indica el tipo de regularización que se aplicará al modelo. En este caso, se especifica 'none', lo que significa que no se aplicará ninguna regularización. Es el que mejor resultados daba.
  - *solver*: es el algoritmo utilizado para optimizar los pesos del modelo. En este caso, se especifica 'lbfgs', que es un algoritmo de optimización basado en el método de gradiente de Broyden-Fletcher-Goldfarb-Shanno. Es el que mejor resultados daba de todos los probados.
  - *class_weight*: es un diccionario o una cadena que indica el peso asignado a cada clase. En este caso, se especifica 'balanced', lo que significa que se asignará un peso inversamente proporcional a la frecuencia de cada clase en los datos de entrenamiento. Esto ayuda a equilibrar la clasificación para clases desequilibradas. De nuevo, es el que mejor resultado daba.

El proceso de entrenamiento del modelo se encuentra en el archivo LogisticRegression.ipynb

Exporta el modelo final a un fichero y reportar F1 score

Al observar la matriz de confusión, se puede ver que el modelo tiene una precisión (*precision*) relativamente alta en la clasificación de los datos. En particular, se observa que el modelo tiene un buen desempeño en la clasificación de los datos como "de pie" y "caminando", ya que la mayoría de las observaciones predichas "de pie" y "caminando" son verdaderamente así. Sin embargo, el modelo tiene un desempeño menor en la predicción de los datos como "escalera", ya que hay un número significativo de predicciones "escalera" que son realmente “de pie”.

Por otra parte, en términos de *recall*, el modelo tiene un desempeño menor en la clasificación de los datos como "de pie", ya que hay un número significativo de observaciones verdaderamente "de pie" que son clasificadas como "escaleras"

```
Confusion Matrix
 [[337   2 226]
 [  0 517  15]
 [ 32   9 436]]

 F1 score : 0.82
```

El F1 Score es una medida de precisión compuesta que tiene en cuenta tanto la *precision* como el *recall*. En este caso, el F1 Score es de 0.82, lo que indica que el modelo tiene un buen desempeño general en términos de *precision* y *recall*.

El modelo final se encuentra en el fichero logistic_regression_model.sav

Haz propuestas de mejora orientadas a llevar F1 score por encima del 95%

Empleo de un Giroscopio

Se podrían integrar datos de un giroscopio en nuestra base de datos. Estos datos nos darían información de la orientación del móvil en los tres ejes de coordenadas para cada uno de los tres estados. Nuestro modelo predictivo podría beneficiarse enormemente de estos datos debido a que las posiciones y velocidades angulares son clave en la identificación de orientaciones del móvil.

Empleo de redes neuronales convolucionales

Nuestro modelo predictivo se podría también beneficiar del uso de redes de convolución, debido a que nuestra base de datos contiene un considerable número de variables. De ahí, se podrían encontrar patrones significativos entre las variables con el empleo de filtros 1D y el uso de modelos de clasificación basados en redes neuronales, lo cual podría aumentar significativamente la precisión de nuestro modelo.

Empleo de validación *cruzada*

El uso de validación cruzada nos ayudaría a evaluar con mejor precisión la calidad y comportamiento de nuestro modelo al ser expuesto a datos nuevos. Adicionalmente, reduciría el riesgo de *overfitting* del modelo.

Filtrado y preprocesado *de datos brutos*

El acelerómetro y otros dispositivos de medición están expuestos a un ruido constante localizado en distintas bandas de frecuencia. Este ruido puede interferir en la calidad de nuestra predicción. Sin embargo, filtrar este ruido (con el empleo de filtros paso banda, filtros de Kalman...) y/o preprocesar los datos brutos *a posteriori* puede resolver este problema y mejorar la calidad de los datos de entrenamiento, lo cual podría aumentar significativamente la precisión de nuestro modelo.

Aprendizaje *en línea*

Se podría desarrollar una aplicación, posiblemente de salud o deportiva, la cual se basaría en predicciones del modelo para proveer información de interés para el usuario. La aplicación recogería datos de uso del usuario, que podrían ser almacenados en una nube y empleados posteriormente para la adaptación y mejoría de nuestro modelo predictivo. De esta forma, cada vez tendríamos un mejor modelo.